SEMI-PERSISTENT RELOCATABLE RAM-BASED VIRTUAL FLOPPY DISK METHOD

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is related to application Serial No. _____ (Att'y. Docket No. 1662-41100) entitled "Reserved ROM Space for Storage of Operating System Drivers," filed concurrently herewith.

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] Not applicable.

BACKGROUND OF THE INVENTION

Field of the Invention

[0003] The present invention relates generally to loading operating systems on computer or server systems. More particularly, the preferred embodiments of the present invention are directed to insuring the availability of operating system drivers during the operating system installation process. More particularly still, the preferred embodiments of the present invention are directed to storing operating system drivers in unreserved ROM and making those drivers available during the installation of the operating system.

Background of the Invention

[0004] One of the first functions that must be performed before a computer or server system is ready for operation is installation of the operating system. In early computer systems, this involved loading the Disk Operating System (DOS) to make available the then-familiar "C:>"

prompt. For many years, DOS was the predominant operating system, even if another form of Graphical User Interface (GUI) was used. In recent years, however, software companies such as Microsoft, Inc. have created and are disseminating software that is not DOS-based. For example, Windows NT is an operating system and GUI in one package. Likewise, Windows 2000 is an operating system and GUI. These new systems communicate with hardware within the computer system by way of operating system or hardware drivers, rather than through traditional basic input/output system (BIOS) routines. During the process of installing these operating systems/user interfaces, it is required that proper drivers are installed for the hardware resident in the computer system.

[0005] Although several software manufacturers make operating systems, *e.g.*, Linux, Novell, and Windows, the problem of installing correct operating system drivers is inherent in each package. In particular, software manufacturers typically bundle, along with the programs that make up their operating system, every operating system driver available as of the release date. However, there may be many months or even years between the release of the operating system and manufacture of the hardware devices within the computer system. Thus, it is inevitable the drivers are needed that are not included with the operating system. Installing an operating system driver other than one of the drivers included with the operating system software typically involves finding the appropriate driver, either on the internet or on a CD ROM included with the computer system. This driver is then typically copied or "punched-out" to a floppy drive (on a second computer as the CD ROM on the computer involved in the installation process is most likely not operational). The floppy disk drive including the required operating system driver is then inserted into the floppy drive unit of the affected computer at the appropriate time during the installation process.

[0006] While it is possible to install the correct operating system driver in this manner, it is seen from the above discussion that this is a complicated procedure. In the context of installing the operating system onto a server in a rack of servers, the situation gets more complex as each individual server may not have its own floppy drive; but rather, keyboard, video and floppy drive access may only be available across a communication bus.

[0007] Thus, what is needed in the art is a way to provide, during the operating system installation, operating system drivers without requiring the user to punch-out floppy disk drives or search the internet via other computer devices to find the necessary drivers.

BRIEF SUMMARY OF THE INVENTION

[0008] The problems noted above are solved in large part by a semi-persistent relocatable RAM-based virtual disk drive. In particular, operating system drivers are provided in ROM space. Those drivers are made available for copying during installation of the operating system by having the drivers appear to reside on a virtual floppy drive in the system.

[0009] In a second aspect of the preferred embodiments, multiple floppy images are preferably stored in the ROM space. Each of the floppy images preferably contains one or more operating system drivers for use with a particular type of operating system. Prior to installation of the operating system, the user preferably selects, by way of BIOS setup screens, which operating system is to be installed on a particular computer or server system. This selection process preferably sets an environment variable within a non-volatile memory in the computer system. BIOS routines, specifically interrupt 13h routines, preferably access the environment variables as an indication of which floppy image to provide in a virtual format such that operating system drivers may be copied at an appropriate time in the installation process.

[0010] In another aspect of the preferred embodiments, the virtual disk drive is provided at any location within the virtual address space. Thus, the actual contents of the virtual disk drive may reside on the ROM, which is mapped in the virtual address space, but also may reside at any other location within the virtual address space, for example within the RAM area of the virtual address space. In this way, a virtual floppy drive having semi-persistent qualities may have its contents placed in the RAM. Such a virtual disk drive contents would survive a warm boot procedure, such as that instituted by an interrupt 19h BIOS routine.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] For a detailed description of the preferred embodiments of the invention, reference will now be made to the accompanying drawings in which:

[0012] Figure 1 shows a computer system of the preferred embodiment;

[0013] Figure 2 shows, in block diagram form, the contents of a ROM of the prior art;

[0014] Figure 3 shows, in block diagram form, the contents of a ROM of one of the embodiments of the present invention;

[0015] Figure 4 shows, in block diagram form, the contents of a ROM of one embodiment of the present invention;

[0016] Figure 5 shows, in block diagram form, the contents of a ROM of one embodiment of the invention, both before and after copying of the operating system drivers;

[0017] Figure 6 shows, in block diagram form, the contents of a ROM of one embodiment of the invention having multiple floppy images contained thereon; and

[0018] Figure 7 shows a virtual memory map of the preferred embodiments.

NOTATION AND NOMENCLATURE

[0019] Certain terms are used throughout the following description and claims to refer to particular system components. As one skilled in the art will appreciate, computer companies may refer to a component by different names. This document does not intend to distinguish between components that differ in name but not function.

[0020] In the following discussion and in the claims, the terms "including" and "comprising" are used in an open-ended fashion, and thus should be interpreted to mean "including, but not limited to...". Also, the term "couple" or "couples" is intended to mean either an indirect or direct electrical connection. Thus, if a first device couples to a second device, that connection may be through a direct electrical connection, or through an indirect electrical connection via other devices and connections.

[0021] Throughout the specification and claims, the term read only memory (ROM) refers to integrated circuit memory devices. If other types of read only memory devices are the focus of the discussion, those will be referred to directly, *e.g.*, compact disk ROM (CDROM) and the like. Thus, the term ROM, unless specifically limited, may mean programmable read only memory (PROM), erasable programmable read only memory (EPROM), and the variants of EPROM such as ultra-violet erasable programmable read only memory (UVPROM) and electrically erasable programmable read only memory (EEPROM).

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0022] Referring now to Figure 1, computer system 100 in accordance with the preferred embodiment comprises at least one CPU 10. Inasmuch as computer system 100 is preferably a server system, the computer system 100 may comprise multiple CPUs 10A, 10B, 10C, 10D

arranged in a configuration where parallel computing may take place. The CPU array 10 couples to a main memory array 12 and a variety of other peripheral computer system components through an integrated host bridge logic device 14. The CPU array 10 may comprise, for example, a plurality of Pentium® III microprocessors. It should be understood, however, that computer system 100 could include other alternative types and numbers of microprocessors. Additionally, other architectures could be used if desired. Thus, the computer system may implement other bus configurations and bus bridges in addition to, or in place of, those shown in Figure 1.

[0023] The main memory array 12 preferably couples to the host bridge logic 14 through a memory bus 16, and the host bridge logic 14 preferably includes a memory control unit (not shown) that controls transactions to the main memory 12 by asserting the necessary control signals during memory accesses. The main memory 12 functions as the working memory for the CPUs 10 and generally includes a conventional memory device or array of memory devices in which program instructions and data are stored. The main memory array 12 may comprise any suitable type of memory such as Dynamic Random Access Memory (DRAM) or any of the various types of DRAM devices such as Synchronous DRAM (SDRAM), Extended Data Output DRAM (EDO DRAM), or Rambus™ DRAM (RDRAM).

[0024] Inasmuch as computer system 100 is preferably a server system, the computer system 100 may not have a dedicated display device. If the computer system did have a dedicated display device, such a system could be implemented by coupling a video driver card to the host bridge 14 by way of an Advanced Graphics Port bus or other suitable type of bus. Alternatively, the video driver card could couple to the primary expansion bus 18 or one of the secondary expansion buses, for example, the PCI bus 20. If the computer system had a dedicated display device, the video

driver or graphic controller would couple to a display device. That display may comprise any suitable electronic display device upon which any image or text can be represented.

[0025] The computer system 100 preferably comprises a second bridge logic device 22 that bridges the primary expansion bus 18 to various secondary buses including a low pin count (LPC) bus 24 and a peripheral component interconnect (PCI) bus 20. In accordance with the preferred embodiment, the bridge device 36 is an Input/Output Controller Hub (ICH) manufactured by Intel Corporation. Although the ICH 22 of Figure 1 is shown only to support the LPC bus 24 and the PCI bus 20, various other secondary buses may be supported by the ICH 22. In the preferred embodiment shown in Figure 1, the primary expansion bus 18 comprises a Hub-link bus which is a proprietary bus of the Intel Corporation. However, computer system 100 is not limited to any particular type of primary expansion bus, and thus other suitable buses may be used.

[0026] Referring still to Figure 1, a firmware hub 26 couples to the ICH 22 by way of the LPC bus 24. The firmware hub 26 preferably comprises Read Only Memory (ROM) which contains software programs executable by the CPU array 10. The software programs preferably include programs to implement basic input/output system (BIOS) commands, and instructions executed during and just after Power On Self Test (POST) procedures.

[0027] A Super Input/Output controller 28 couples to the ICH 22 and controls many system functions including interfacing with various input and output devices such as keyboard 30. The Super I/O controller 28 may further interface, for example, with a system pointing device such as a mouse 32, various serial ports (not shown) and floppy drives (not shown). The Super I/O controller is often referred to as "super" because of the many I/O functions it may perform.

[0028] Also shown in the computer system 100 of Figure 1 are three array controllers 50A, 50B, 50C coupled to the ICH 22 by way of the PCI bus 20. Each array controller 50 couples to a

plurality of hard drives 52A, 52B, 52C. Thus, the array controller 50 preferably performs data reads, data writes and other necessary data manipulation to implement a redundant array of independent devices (RAID) system. It will be understood that while Figure 1 shows only three array controllers 50, computer system 100 may support any number of these controllers. It must be understood however that the invention is not limited to computer systems having multiple CPUs or implementing RAID systems; rather, the preferred embodiments apply equally to all types of computer systems.

[0029] The preferred embodiments of the present invention address how to provide operating system drivers during the operating system installation process. In particular, the preferred embodiments of the present invention provide necessary operating system drivers by placing or storing those drivers in the ROM or firmware hub 26. Preferably, required or needed operating system drivers may simply be copied from those versions resident in the system ROM 26 during the operating system installation procedure.

[0030] In computer systems requiring high availability and reliability, *e.g.*, server systems, it is common to have multiple copies of the BIOS firmware or programs stored in the system ROM 26. Figure 2 shows a prior art technique for having multiple copies that simply involves burning two copies of the BIOS programs onto the ROM 26. In particular, Figure 2 shows two copies of the BIOS, with the first copy occupying the first 512 kilobytes of the ROM, and the second copy occupying the second 512 kilobytes of the one megabyte ROM. As one of ordinary skill in the art is aware, most computer and server systems have a smaller ROM known as a "boot-block" ROM (not shown) that is responsible for selecting which of the multiple copies of the BIOS will be loaded during the POST procedure.

[0031] Placing operating system drivers in the ROM 26 may take many forms. Figure 3 shows one embodiment in which the operating system drivers (labeled OSD in the drawing) are placed within each redundant section of the ROM 26. At the appropriate time during the operating system installation procedure, either or both of the drivers stored on the ROM 26 would be made available for copying and use by the operating system. A preferred embodiment for making those operating system drivers available is discussed more fully below.

[0032] Operating system drivers range in complexity, and therefore range in size, with some drivers approaching 100 kilobytes or more. In a situation where the drivers are large, the implementation shown in Figure 3 would not be desirable inasmuch as the operating system drivers would occupy significant space in the ROM (especially given the duplication). Conversely, if the operating system drivers to be provided on the ROM 26 are relative small, the implementation exemplified in Figure 3 may be acceptable.

[0033] Where the operating system drivers are large, or a number of drivers must be provided, the preferred embodiments provide for storing those operating system drivers in a non-redundant area of the ROM 26. Figure 4 shows an implementation where a larger, preferably two megabyte, ROM is used. Preferably, the ROM is divided up into redundant 30 and non-redundant 32 space. In Figure 4, the redundant space is shown to occupy the first one megabyte of the ROM 26, and the non-redundant space, containing the operating system drivers, is shown to occupy the second one megabyte. It must be understood, however, that division of the ROM 26 of Figure 4 is only exemplary, and the storage space of the ROM 26 may be divided in any convenient way. Using the implementation exemplified in Figure 4, a boot-block program stored on a boot-block ROM (not shown), selects one of the two BIOS firmware copies stored in the redundant area 30 of the ROM 26. Thereafter, and at the appropriate time during the operating system installation

procedure, the operating system drivers located in the non-redundant area 32 of the ROM are made available.

[0034] In the exemplary implementation of storing operating system drivers on the ROM 26 as shown in Figure 4, the size of the ROM 26 was increased from one megabyte to two megabytes, such that the duplicate copies of the BIOS and the operating system drivers could reside on a single ROM device. However, increasing the size of the ROM 26 also increases system cost. For this reason, some manufacturers may not want to spend the additional money to provide the larger ROMs, but still may want to provide redundant copies of the BIOS and also provide copies of the operating system drivers on the ROM.

[0035] Figure 5 shows one possible implementation for providing both the redundant BIOS and the operating system drivers on the ROM. In particular, the upper ROM 26 is shown to have a single copy of the BIOS programs, as well as a copy of the operating system drivers. Preferably, the ROM 26 has this configuration as it leaves the factory and during the operating system installation procedure. The program stored in the boot block ROM (not shown) determines whether the ROM contents are BIOS programs or operating system drivers by use of signatures in the BIOS firmware that identify the programs. Preferably, once the operating system drivers have been provided during installation of the operating system, a utility program copies the BIOS, provided only in non-redundant fashion initially, to the second half of the ROM 26. By copying the BIOS over the operating system drivers, a redundant BIOS system is provided, as shown in the lower ROM 26 of Figure 5. While this implementation provides both the redundant BIOS, after operating system installation, and also provides operating system drivers on the ROM, the operating system drivers are overwritten and thus will not be available if the operating system must be installed again.

[0036] While there may be many ways to implement making the operating system drivers stored on the ROM 26 available during the operating system installation procedure, the Applicants now endeavor to describe the preferred method. Making the operating system drivers available has three aspects in the preferred embodiment: 1) providing the latest drivers for each major operating system; 2) making those drivers available to the user during installation of the operating system by use of a virtual disk drive; and 3) providing those operating system drivers in a virtual drive scheme with the system drivers residing in RAM.

[0037] As discussed in the Background section, there are many available operating systems for computers in the marketplace, *e.g.*, Linux (manufactured by Red Hat Software), Novell (manufactured by Novell Incorporated), Windows 2000 (manufactured by Microsoft Inc.). For a particular piece of hardware in the system, *e.g.*, an array controller 50 (Figure 1), each operating system may require a different driver. This may be due in part to differences in protocols with regard to interfacing with the operating system, but may also be caused merely by differences in file structures between operating systems. Stated otherwise, a driver for operating an array controller 50 in a Windows 2000 environment (FAT file system) may not be suitable for use in a Linux environment (2 x 72 file system). Thus, in the preferred embodiment, if drivers need to be included on the system ROM 26, preferably drivers for each of the major operating systems are provided. Considering that there may be an array of devices for which the latest operating system driver may be provided, it is easily seen that the amount of space on the ROM 26 required to store each driver for each hardware system may become rather large. In this regard, the implementation shown in Figure 4 for dividing the ROM 26 into a redundant 30 and non-redundant 32 portion is the preferred implementation.

[0038] Figure 6 exemplifies the situation where multiple sets of operating system drivers are provided. In particular, Figure 6 shows that the non-redundant 32 space of the ROM 26 contains several sets of drivers, in the exemplary system, a set of Linux drivers 34, Novell drivers 36 and Windows drivers 38. It must be understood that providing these particular drivers is only exemplary, and any number of drivers for any number of operating systems may be provided and still be within the contemplation of this invention. Further, Figure 6 implies that the operating system drivers occupy the entire non-redundant 32 space of the ROM 26; however, this need not necessarily be the case, and any or all of the non-redundant 32 space may be used to store the operating system drivers.

[0039] In the preferred embodiment described above, a set of operating system drivers (34, 36, 38 of Figure 6), which may alternatively be referred to as floppy images, are provided in the ROM 26. These multiple sets of operating system drivers are provided to account for the fact that a manufacturer may not know at the time of building the computer system what operating system will be installed thereon. Since multiple sets of operating system drivers are preferably provided, there should be a method of providing the correct drivers for the particular operating system. In the preferred embodiments, making available the correct operating system drivers for the operating system is preferably accomplished by a BIOS setup parameter and a modification to the standard disk access routine known as Interrupt 13h.

[0040] In the preferred embodiment the BIOS setup routines are modified to contain a field where a user, when making an initial setup of the BIOS, selects which operating system is to be installed on the computer or server. It must be understood that this selection in the BIOS is not a part of the operating system installation procedure; but rather, is merely a mechanism to inform the BIOS which operating system is to be installed. Preferably, the BIOS uses this information to

select an appropriate floppy image having a set of operating system drivers from the images stored in the non-redundant 32 space of the ROM 26. For example, if a user selects from the BIOS setup screen that Windows 2000 will be the operating system for the computer, the BIOS then makes available, in a manner described more fully below, the floppy image containing Windows operating system drivers 38 (see Figure 6).

[0041] While there may be many ways to make the particular operating system drivers available during the operating system installation process, in the preferred embodiments, those drivers are made available to the user and to the operating system installation procedure by having them reside on a virtual floppy drive or virtual disk drive. More particular, in the preferred embodiment, the Interrupt 13h BIOS calls for performing disk drive activities are preferably implemented such that the operating system drivers (either all of them or just the appropriate drivers for the particular operating system) stored on the ROM 26 appear to reside on a disk drive. Because the files are not actually stored on a floppy disk, this is known as creating a virtual drive. As mentioned above, the user preferably selects in a BIOS setup screen which operating system is to be installed on the computer system, and based on that selection, preferably only the drivers appropriate for the selected operating system are made available in this virtual drive method. One of ordinary skill in the art is familiar with Interrupt 13h BIOS calls, how to implement them, and now understanding how to provide drivers in this way, could modify the standard programs to provide this virtual drive feature.

[0042] More particularly still, in the preferred embodiment, selecting a particular operating system to be installed preferably sets an environment variable in a non-volatile memory. This non-volatile memory could be non-volatile RAM (NVRAM), or may be written directly to the ROM 26, which in the preferred embodiment is electrically erasable programmable read only memory

(EEPROM). Regardless of the location of the environment variables, by selecting an operating system, the environment variables preferably point to a floppy image having operating system drivers appropriate for that operating system. During the installation process, before the operating system is installed, disk services are provided by the BIOS. Thus, during installation, the Interrupt 13h services preferably are adapted to show the appropriate operating system drivers, indicated by the environment variables, as residing on a disk drive. Thus, drivers needed for correct setup during the operating system installation process are then available as if they had been copied to a floppy and inserted in a disk drive in the system. Using the preferred embodiment alone, it is then possible to provide the operating system drivers during the operation system installation process by informing the installation program of the need to use drivers different than those provided with the operating system, and pointing that operating system software to the virtual drive. However, now understanding how to make available software drivers in the manner of the preferred embodiment, one of ordinary skill in the art could easily modify the installation program to automatically search for and use drivers resident on virtual drives with little or not user input.

[0043] Referring now to Figure 7, there is shown an exemplary virtual address space or virtual memory map for the computer or server system of the preferred embodiment. As one of ordinary skill in the art is aware, the ROM 26 contents are typically mapped in the virtual address space to a location having addresses just below the four gigabyte addressable space. In this way, the system devices wishing to access the ROM contents need only know the addresses of those ROM contents in the virtual address space. As exemplified in Figure 7, the virtual address space includes not only the floppy images having the operating system drivers 34, 36 and 38, but also includes all the addressable space for the random access memory (RAM), the RAM area. Thus, in the preferred embodiment, when the user selects the particular operating system to be installed in the BIOS setup

screens, the BIOS preferably updates an environment variable which points, in the virtual address space, to the operating system drivers appropriate for that operating system to be installed. Sometime thereafter, when performing disk accesses using the BIOS subroutines, the installation process requests an operating system driver from a virtual drive. The Interrupt 13h services provide the operating system drivers available at the location indicated by the environment variables as if those drivers were provided on an actual floppy drive.

[0044] It is noted, however, that the mapping of the possible operating system drivers 34, 36 and 38 are within the same virtual address space as the random access memory (Figure 7). In another aspect of the invention, information and data may be provided to the operating system by use of a virtual disk drive, where the contents of that virtual disk drive resides in the RAM area 62 portion of the virtual memory rather than one of the operating system drivers, again within the virtual address space. Data and programs placed in a virtual drive with its contents stored in RAM would be semi-persistent, meaning that the data would survive and be available after a warm boot, for example an warm boot initiated by an Interrupt 19h.

[0045] While creating a virtual disk drive resident in RAM and accessible through BIOS interrupts was developed in the context of providing operating system drivers during operating system installation procedures, any data or programs which need to be made available in a semi-persistent manner may be provided in this way, and still be within the contemplation of the invention.

[0046] The above discussion is meant to be illustrative of the principles and various embodiments of the present invention. Numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. For example, the preferred embodiments have been described with regard to placing the operating system drivers on

the same ROM as the BIOS programs; however, the ROM provided in the computer system could be a ROM array comprising several individual ROM devices, and the operating system software could reside on any or all of the ROM devices, and still be within the contemplation of this invention. Further, providing operating system drivers in the manner described herein was developed in the context of server systems having multiple microprocessors and multiple hard drives implementing RAID systems; however, one of ordinary skill in the art, now understanding how to implement and use the preferred embodiments, could easily apply the preferred embodiments to stand alone computer systems having only a single microprocessor and single hard drive. Further, the systems and methods described herein are equally applicable to most computing devices such as hand-held computing devices, portable computers, process control systems, and the like. It is intended that the following claims be interpreted to embrace all such variations and modifications.